Annales Univ. Sci. Budapest., Sect. Comp. 49 (2019) 97–116

LOAD BALANCING IN DISTRIBUTED POINT CLOUD DATABASES

István Csabai, László Dobos,
Attila Kiss and János M. Szalai-Gindl
(Budapest, Hungary)

Communicated by András Benczúr
(Received March 15, 2019; accepted July 10, 2019)

Abstract. There are several approaches related to handling and storing massive amounts of multidimensional data. State-of-the-art database systems use shared-nothing architecture for scalable spatial data management and indexing where data co-location and query load balancing are the primary objectives. Hence, data placement is an important component of efficiency. Szalai-Gindl et al. (SG17) proposed a data distribution algorithm for this task in an earlier work [26]. This paper investigates the improvement possibilities of that algorithm.

1. Introduction

The focus of our work is to investigate how to handle and store massive amounts of static, multidimensional point clouds from scientific research where the distribution of data is organized hierarchically or otherwise heavily unbalanced or skewed. Examples of such categories include point clouds (such as LiDAR data, cosmological N-body simulations, intersections of road networks etc.) or objects from different regions in OpenStreetMap (OSM) which

Key words and phrases: Multi-dimensional histogram partitioning, data placement, load balancing.
2010 Mathematics Subject Classification: 68M14.
The project has been supported by the Hungarian National Research, Development and Innovation Office under grant no. OTKA NN 114560, OTKA NN 129148 and by the European Union, co-financed by the European Social Fund (EFOP-3.6.3-VEKOP-16-2017-00002).

are not evenly distributed [29]. State-of-the-art database systems follow the shared-nothing paradigm [1] comprising of a set of independent nodes where each node has its own CPU, RAM and disk and the nodes can communicate via high bandwidth network. We investigate how data can be distributed efficiently on such architectures. For simplicity, we use the word 'server' and 'node' interchangeably.

When working with massive amounts of data, it is important to optimize spatial query processing by using distributed spatial database systems. To this end, indexing and access methods supporting multidimensional data are necessary. Over the last 40 years, numerous algorithms were proposed to address these problems for single machine set ups. Detailed research of the literature shows, however, that only a few work dealt with distributed environments, as it is emphasized by Samet [24] in his classic book on spatial data structures. More recently, researchers have been paying particular attention to the study on how to redesign existing storage models and algorithms for distributed for Big Data platforms [32].

There are two important types of spatial queries: box- and nearest neighbor queries. According to [16], it would be useful to define two requirements in connection with distributed search trees: 'minLoad' and 'uniSpread'. The requirement 'minLoad' means that a localized query affects as few index tree nodes as possible during the evaluation process. The advantage of this is that queries with small search regions activate as few servers as possible. The requirement 'uniSpread' means that the tree nodes that are affected by a query are uniformly distributed among servers. As a consequence, queries with large result sets activate as many servers as possible. In case of using an R-tree, the trade-offs among planning decisions were examined and discussed in an early paper [3], although on different assumptions and requirements than here. In [3] an important concept called 'distribution policy' was introduced which determined what aspects of query optimization should dominate data distribution among servers. 'Clustering' policy organizes data across servers in such a way that data locality is preserved. This contributes to the fulfillment of 'minLoad' which supports queries with small region of interest and nearest neighbor search. Contrary, 'declustering' policy attempts to optimize for 'uniSpread' to accelerate queries with large result set. 'Balance' policy distributes data points among the servers so that data is evenly distributed and no particular query type has a prominent role.

In our earlier work [26], we investigated data placement in distributed environments with the goal of optimizing spatial queries. A 'load balancing' algorithm (SG17) was introduced which supported the 'balance' data distribution policy only. In [10], we proved that if data points span a sufficiently high dimensional space, then the aforementioned algorithm also optimizes the response times of box queries, where query boundaries are parallel to the co-

ordinate system. Note, that we use the phrases 'box query' and 'range query' interchangeably. In this work we derive formulas for upper limits of absolute deviation of server weights which can be utilized to accelerate the load balancing algorithm.

This paper is organized as follows. Sec. 2 provides details about related work. The SG17 algorithm is described in Sec. 3. Upper limit formulas for the absolute deviation of server weights are derived in Sec. 4. Finally, Sec. 5 contains the conclusions and future work.

2. Related work

Following the classification scheme of [6] of point access methods, one can distinguish Data Partitioning (DP)-based and Space Partitioning (SP)-based index structures. In [6], the authors emphasize that neither is really suitable for high dimensional data. In the case of DP-based search trees, such as R-trees, the problem is the high degree of overlapping between minimum bounding boxes (MBB). In case of static data, however, the issue can be mitigated if the nearby points are cleverly grouped by a clustering algorithm or spatial division and their MBBs are stored at the leaf level. For example, [13] provides a method which utilizes an octree to subdivide the space and R-tree leaf nodes are generated by direct insertion of neighboring points from the octree subdivisions.

The majority of related papers discussing distributed environments [16, 17, 25, 3, 22, 31, 11, 28] focus on indexing methods of the R-tree family. Applications of space filling curves (SFCs) for parallel domain decomposition and load balancing can be found, for example, in [2] and [27]. Grid-based techniques are investigated by [20, 4, 8]. Some approaches exist which utilize kd-tree for space partitioning on top of local index structures [22, 23]. In [9], a distributed high-dimensional index structure (DVA-tree) is proposed which is based on a hybrid spill-tree and Vector Approximation files. Some papers [18, 12] make the assumption that prior information is available, based on past experience, logs, etc. on the distribution of query sets. Previous work mainly focused on declustering and box queries [16, 17, 20, 25, 4, 8, 18] or clustering and proximity search [9, 23]. However, there were also attempts to satisfy these conflicting demands [3, 22, 27, 31, 12]. Methods which are related to Big Data platforms [21, 33, 11, 28] usually contain little novelty from a theoretical point of view, although they are very important in practice.

Contrary to the related works, we want to utilize a priori information about the spatial distribution of the data in the form of a multidimensional histogram which can be obtained during the data extraction or transformation phase of a typical Extraction-Transformation-Loading (ETL) process. This approach was also followed by our earlier papers [26, 10]. Although based on a histogram

technique, the load balancing algorithm investigated here can also be applied in cases when, instead of binning, multiple data points are represented by centroids derived from clustering or spatial division. Applying the algorithm to cluster centers would alter the distribution policy from 'balance' to 'clustering'. In the storage layer, one can use a generic two-tier index scheme including global and local index structures. The global index structure is built on the multidimensional histogram and is maintained at a central server. The local index structures are created when point groups (or bins) are distributed across slave servers. Theoretically, spatial index structures are arbitrary.

3. Load balancing

Let us consider the continuous $\mathcal{D}$-dimensional vector space with the usualEuclidean norm. We turn our attention to a subset $B \subset \mathbb{R}^{\mathcal{D}}$ which is the bounding box of r data points. We compute the histogram of point counts within B with regular, equally-spaced binning in Cartesian coordinates. Note, that many of our conclusions remain true for the case of less regular or irregular binning. It is necessary, however, to compute the histogram quickly during the data extraction or transformation phase of an ETL procedure. There are $n^{\mathcal{D}}$ bins: $\{B_1, \ldots, B_{n^{\mathcal{D}}}\}$ where n is the resolution of the histogram. We denote by $w(B_j)$ the number of points falling into bin B_j. We use the term 'weight' to refer to the point count in a particular bin further in the text. The $r = \sum_j w(B_j)$ data points will be apportioned among s servers: $S_1, S_2, \ldots, S_s$ ($s \geq 2$). The design guarantees that points within a bin are stored on the same server. The $T(B_j, S_i)$ binary assignment function will take the value of 1 if bin B_j is associated with server S_i, otherwise will be set to 0. By the assignment of bins we mean the set of pairs (B_j, S_i) where $T(B_j, S_i) = 1$. Let us denote by $w(S_i)$ and $b(S_i)$ the number of data points (weight) and the number of bins, respectively, belonging to server S_i. The mean weight of servers is defined simply as $\bar{S} = r/s$, while the bin with maximum weights will be referred to as $B_{\max}$.

We can elaborate the assignment as a (0-1)-integer programming problem for the domain of the binary assignment function $T(B_j, S_i)$ if the sole aim is to ensure load balancing. There are many possibilities to fulfill this. On one hand, one can demand that a bin group is connected or specially shaped (e.g. hyperrectangle). On the other hand, one can optimize for load balancing in different ways: for example, minimizing the maximum of the server weights or minimizing the mean absolute deviation of weights. We choose the latter option. Additionally, we will not require connectedness. The really interesting constraint is that all the bins have to be assigned to exactly one of the servers, i.e. $\sum_{i=1}^{s} T(B_j, S_i) = 1$ for all j. Consequently, $w(S_i) = \sum_{j=1}^{n^{\mathcal{D}}} T(B_j, S_i) w(B_j)$. We would like to minimize $\sum_{i=1}^{s} |w(S_i) - \bar{S}|$ to achieve the best load balancing.

The minimization of the sum of absolute deviations can be transformed into a linear programming problem, see [7, 30]. Write $d_i = w(S_i) - \bar{S}$ and $d_i^{\text{abs}} = |d_i|$.* Using these notations, the optimization problem becomes the following:

$$\begin{aligned}
\text{minimize} \quad & \sum_{i=1}^{s} d_i^{\text{abs}} \\
\text{subject to} \quad & d_i + \sum_{j=1}^{n^{\mathcal{D}}} w(B_j) \cdot T(B_j, S_i) = \bar{S}, \\
& d_i \le d_i^{\text{abs}}, \ -d_i \le d_i^{\text{abs}}, \\
& \sum_{i=1}^{s} T(B_j, S_i) = 1 \text{ for all } j, \\
& T(B_j, S_i) \in \{0, 1\}, \\
& i = 1, ..., s; \ j = 1, ..., n^{\mathcal{D}},
\end{aligned}$$

To this end, we only recall the SG17 algorithm of [26]. However, one can add an additional constraint to the optimization problem since Theorem 4.1 of Sec. 4.1 provides a general upper limit of server weights for optimal load balancing. Therefore, one can replace the $d_i \le d_i^{\text{abs}}$ and $-d_i \le d_i^{\text{abs}}$ constraints by $d_i \le \bar{S}$ and $-d_i \le \bar{S}$. Even stricter constraints can be used if prior knowledge about data distribution is available (see the paragraph following Eq. 4.1). The advantage of using these constraints is that the running time of the algorithm can be improved.

4. Upper limits of absolute deviation of server weights

In this section, we find a constant c for the following inequality in the general case and in a special case where data points are sparse but strongly clustered on a wide range of scales (including cosmological N-body simulations and intersections of road networks):

$$|w(S_i) - \bar{S}| \le c \cdot \bar{S}. \tag{4.1}$$

In the general case, c is equal to 1 (see Corollary 4.1). Furthermore, we describe a method in Alg. 1 which can be utilized to create actual bin assignments. Theoretically and experimentally, we investigate the results of this approach in the above-mentioned special case (see Sec. 4.3) and conclude that c can be less than 1 with a high degree of likelihood for 'real-world' data.

*Cited articles introduce slightly different auxiliary variables. The connections between them are $d_i^{\text{abs}} = d_i^+ + d_i^-$ and $d_i = d_i^+ - d_i^-$ where $d_i^+, d_i^- \ge 0$ constraints are equivalent to $d_i \le d_i^{\text{abs}}, -d_i \le d_i^{\text{abs}}$.

Algorithm 1: Bin assignment method

input : bin weights: $\{w(B_j)\}_{j=1}^{n^{\mathcal{D}}}$, the number of servers: s
output: an assignment of bins to servers

sort the set $\{w(B_j)\}_{j=1}^{n^{\mathcal{D}}}$
renumber the bins based on sorting: $0 \leq w(B_1) \leq \cdots \leq w(B_{n^{\mathcal{D}}})$
for $j \leftarrow 1$ **to** $n^{\mathcal{D}}$ **do**
  $i = (j - 1 \pmod s) + 1$
  assign bin B_j to server S_i
end

4.1. General upper limit of server weights

The aim of this subsection is to derive a general formula for the upper limit of server weights:

Theorem 4.1. *Suppose that $w(B_{\max}) \leq \bar{S}$. There exists an improvement for an arbitrary assignment of bins such that*

$$w(S_i) \leq 2 \cdot \bar{S} \text{ for all } i = 1, \ldots, s \tag{4.2}$$

holds.

One can make the assumption that $w(B_{\max}) \leq \bar{S}$ without loss of generality because it can be achieved by increasing the resolution. The proof consists of the construction of an 'augmenting' algorithm which can improve an arbitrary assignment of bins in such a way that Eq. 4.2 holds (see Sec. 4.2). It can be easily shown that this bound is asymptotically sharp as s tends to ∞, i.e. bin weights exist such that $\max_i w(S_i)/(2\bar{S}) \to 1$ as $s \to \infty$. Consider the following example: suppose $s = r - 1$, $w(S_i) \geq 1$ for all $i = 1, \ldots, s$ and $w(B_j) \leq 1$ for all $j = 1, \ldots, n^{\mathcal{D}}$. (Again, the third assumption can be satisfied by increasing the resolution.) Then there is a server S for which $w(S) = 2$ by using the pigeonhole principle and $w(S)/(2\bar{S}) = 2/(2 + 2/s) \to 1$ as $s \to \infty$.

Let us now state a corollary of Theorem 4.1.

Corollary 4.1. *Suppose that $w(B_j) \leq \bar{S}$ $(j = 1, \ldots, n^{\mathcal{D}})$. Then for an optimal assignment of bins:*

$$|w(S_i) - \bar{S}| \leq \bar{S}, \tag{4.3}$$

$$\sigma_s \leq \bar{S}, \tag{4.4}$$

where $i = 1, \ldots, s$ and σ_s denotes the standard deviation of $w(S_1), \ldots, w(S_s)$.

Proof. Using the facts that $0 \leq w(S_i)$ and $w(S_i) \leq 2 \cdot \bar{S}$, which are equivalent to $\bar{S} - w(S_i) \leq \bar{S}$ and $w(S_i) - \bar{S} \leq \bar{S}$, Eq. 4.3 is established. Hence

$\left(w(S_i) - \bar{S}\right)^2 \leq \bar{S}^2$. Consequently, the following inequality holds for the variance of $w(S_1), \ldots, w(S_s)$:

$$\sigma_s^2 = \frac{1}{s} \cdot \sum_{i=1}^{s} \left(w(S_i) - \bar{S}\right)^2 \leq \frac{1}{s} \cdot \sum_{i=1}^{s} \bar{S}^2 = \bar{S}^2,$$

and the proof is complete. ■

Without loss of generality we can assume that $0 < w(S_1) \leq \cdots \leq w(S_s)$ and let $0 \leq K_i$ be an integer for which $\left\lceil \frac{w(S_i)}{\bar{S}} \right\rceil = K_i + 1$ (i.e. $K_i \cdot \bar{S} < w(S_i) \leq (K_i + 1) \cdot \bar{S}$). Furthermore, we can certainly assume that $2 \leq K_s$, since otherwise there would be no need for the improvement. It is clear that if $3 \leq K_s$, then

$$\left\lfloor \frac{K_s}{2} \right\rfloor + 2 \leq K_s. \tag{4.5}$$

Since $s \cdot \bar{S} = r = \sum_{i=1}^{s} w(S_i) > \sum_{i=1}^{s} K_i \cdot \bar{S}$, therefore

$$\sum_{i=1}^{s} K_i \leq s - 1 \tag{4.6}$$

Let us denote by s_0 the number of servers for which $K_i = 0$ (and $s_0 < s$ because $2 \leq K_s$). The following simple lemma is required for the proof of Theorem 4.1.

Lemma 4.1. *$K_s \leq s_0$ holds.*

Proof. Using Eq. 4.6 and the fact that $1 \leq K_i$ if $i = s_0 + 1, \ldots, s - 1$, we get:

$$\begin{aligned} \sum_{i=1}^{s} K_i = \sum_{i=1}^{s_0} K_i + \sum_{i=s_0+1}^{s} K_i &\leq s - 1, \\ (s-1) - (s_0 + 1) + 1 \leq \sum_{i=s_0+1}^{s-1} K_i &\leq s - 1 - K_s, \\ K_s &\leq s_0. \end{aligned}$$

■

4.2. Proof of Theorem 4.1

To prove Theorem 4.1, we use mathematical induction on K_s. First, we consider $K_s = 2$. Then we can rewrite $w(S_s)$ as

$$w(S_s) = 2 \cdot \bar{S} + \epsilon_s \tag{4.7}$$

where $1 \leq \epsilon_s \leq \bar{S}$. Suppose that bins $B_1, \ldots, B_{b(S_s)}$ are assigned to server S_s thus $w(S_s) = w(B_1) + \cdots + w(B_{b(S_s)})$. There exists an integer k such that $w(B_{k+1}) + \cdots + w(B_{b(S_s)}) \leq \bar{S} + \epsilon_s$ but $w(B_k) + \cdots + w(B_{b(S_s)}) > \bar{S} + \epsilon_s$. The bins $B_{k+1}, \ldots, B_{b(S_s)}$ remain on server S_s therefore

$$w(S_s^{\text{new}}) = w(B_{k+1}) + \cdots + w(B_{b(S_s)}) \leq \bar{S} + \epsilon_s \leq 2 \cdot \bar{S}.$$

Using Lemma 4.1, there are at least two servers with such weights which are not greater than $\bar{S}$. The bin B_k is reassigned to one of them whose weight is $w(B_k) \leq \bar{S}$, thus the new server weight is not greater than $2 \cdot \bar{S}$. The bins $B_1, \ldots, B_{k-1}$ are reassigned to the other server (if $k > 1$, otherwise none of the bins are reassigned). The following is true:

$$w(B_1) + \cdots + w(B_{k-1}) < \bar{S}, \tag{4.8}$$

because, on the contrary, suppose $w(B_1) + \cdots + w(B_{k-1}) \geq \bar{S}$. As $w(B_k) + + \cdots + w(B_{b(S_s)}) > \bar{S} + \epsilon_s$, thus we could find $w(B_1) + \cdots + w(B_{b(S_s)}) > 2 \cdot \bar{S} + \epsilon_s$, contrary to Eq. 4.7. The weight of this server is less than $2 \cdot \bar{S}$ with additional bin weights because of Eq. 4.8. As the weights of all concerned servers are not greater than $\bar{S}$ after the reassignment, the number of servers for which the weights are greater than $2 \cdot \bar{S}$ has been reduced by one. The servers are renumbered based on the new weights and one can repeat the above procedure. This is continued until no further server is found for which $w(S_i) > 2 \cdot \bar{S}$. (The algorithm terminates because there are a finite number of servers and at every step there is at least one server whose weight drops below $2 \cdot \bar{S}$.)

Now suppose $K_s > 2$. Then, in accordance with the previous notation:

$$K_s \cdot \bar{S} + \epsilon_s = w(S_s) = w(B_1) + \cdots + w(B_{b(S_s)}) \tag{4.9}$$

where $1 \leq \epsilon_s \leq \bar{S}$. There exists k_1 such that

$$w(B_{k_1+1}) + \cdots + w(B_{b(S_s)}) \leq 2 \cdot \bar{S}, \text{ but} \tag{4.10}$$

$$w(B_{k_1}) + \cdots + w(B_{b(S_s)}) > 2 \cdot \bar{S}. \tag{4.11}$$

The following is true:

$$w(B_{k_1+1}) + \cdots + w(B_{b(S_s)}) \geq \bar{S}, \tag{4.12}$$

because, on the contrary, suppose $w(B_{k_1+1}) + \cdots + w(B_{b(S_s)}) < \bar{S}$. As $w(B_{k_1}) \leq \leq \bar{S}$, thus we could find $w(B_{k_1}) + \cdots + w(B_{b(S_s)}) < 2 \cdot \bar{S}$, contrary to Eq. 4.11. Using Eq. 4.10 and Eq. 4.12, $0 \leq 2 \cdot \bar{S} - (w(B_{k_1+1}) + \cdots + w(B_{b(S_s)})) \leq \bar{S}$ holds. Using this and rearranging Eq. 4.9 gives

$$\begin{aligned} w(B_1) + \cdots + w(B_{k_1}) = &(2 \cdot \bar{S} - (w(B_{k_1+1}) + \cdots + w(B_{b(S_s)}))) + \\ &+ (K_s - 2) \cdot \bar{S} + \epsilon_s \end{aligned} \tag{4.13}$$

$$(K_s-2)\cdot\bar{S}+\epsilon_s\le w(B_1)+\cdots+w(B_{k_1})\le \le \bar{S}+(K_s-2)\cdot\bar{S}+\epsilon_s=(K_s-1)\cdot\bar{S}+\epsilon_s \tag{4.14}$$

If $K_s-1=2$, then this is the case $K_s=2l+1$ which is detailed below. Otherwise if $K_s-1>2$, then using Eq. 4.14 and the fact that $w(B_j)\le\bar{S}$ $(j=1,\ldots,n^{\mathcal{D}})$, there exists an integer k_2 for the remainder weight $w(B_1)+$ $+\cdots+w(B_{k_1})$ such that

$$w(B_{k_2+1})+\cdots+w(B_{k_1})\le 2\cdot\bar{S}, \text{ but} \tag{4.15}$$

$$w(B_{k_2})+\cdots+w(B_{k_1})>2\cdot\bar{S}. \tag{4.16}$$

As $w(B_{k_2})\le\bar{S}$, thus $w(B_{k_2+1})+\cdots+w(B_{k_1})\ge\bar{S}$ otherwise there is a contradiction with Eq. 4.16. As a result of the above considerations:

$$(K_s-4)\cdot\bar{S}+\epsilon_s\le w(B_1)+\cdots+w(B_{k_2})\le(K_s-3)\cdot\bar{S}+\epsilon_s \tag{4.17}$$

This is continued until the coefficient of $\bar{S}$ is negative on the left side of the previous inequality (4.17), i.e. we want to find l which is the largest among such integers for which $K_s-2l\ge 0$. This is $l=\left\lfloor\frac{K_s}{2}\right\rfloor$.

If $K_s=2l+1$, then Eq. 4.17 is

$$\bar{S}+\epsilon_s\le w(B_1)+\cdots+w(B_{k_l})\le 2\cdot\bar{S}+\epsilon_s, \tag{4.18}$$

thus, using $w(B_j)\le\bar{S}$ $(j=1,\ldots,n^{\mathcal{D}})$, there exists an integer k_{l+1} such that

$$w(B_{k_{l+1}+1})+\cdots+w(B_{k_l})\le\bar{S}+\epsilon_s, \text{ but} \tag{4.19}$$

$$w(B_{k_{l+1}})+\cdots+w(B_{k_l})>\bar{S}+\epsilon_s. \tag{4.20}$$

Then $w(B_1)+\cdots+w(B_{k_{l+1}-1})<\bar{S}$ (otherwise combining it with Eq. 4.20, there is a contradiction with Eq. 4.18).

If $K_s=2l$, then Eq. 4.17 is

$$w(B_1)+\cdots+w(B_{k_l})\le\bar{S}+\epsilon_s(\le 2\cdot\bar{S}). \tag{4.21}$$

There is no more division to do, unlike in the case of $K_s=2l+1$.

The bins $B_{k_1+1},\ldots,B_{b(S_s)}$ remain on server S_s. The further bins which are originally assigned to server S_s are grouped:

$$\{B_{k_2+1},\ldots,B_{k_1}\},\ldots,\{B_1,\ldots,B_{k_l}\}$$

in the case of $K_s=2l$ and the last group are subdivided into

$$\{B_{k_{l+1}+1},\ldots,B_{k_l}\},B_{k_{l+1}},\{B_1,\ldots,B_{k_{l+1}-1}\}$$

in the case of $K_s = 2l + 1$, respectively. Therefore the number of groups is $l + 2$ in odd case and l in even case, respectively. For these groups, it is true that their weights are less than or equal to $2 \cdot \bar{S}$. (Moreover, for groups $B_{k_{l+1}}$, $\{B_1, \ldots, B_{k_{l+1}-1}\}$ it is true that their weights are less than or equal to $\bar{S}$.)

Lemma 4.1 and Eq. 4.5 yield $l + 2 \leq K_s \leq s_0$, thus there are at least $l + 2$ servers for which $w(S_i) \leq \bar{S}$. Group $\{B_{k_2+1}, \ldots, B_{k_1}\}$, group $\{B_{k_3+1}, \ldots, B_{k_2}\}$, etc. are reassigned to one of them. After reassignment the new K_i is certainly less than K_s for all $i = 1, \ldots, s$. Therefore, the number of servers for which the weights are greater than $K_s \cdot \bar{S}$ has been reduced. The servers are renumbered based on the new weights and this procedure is continued until no further server is found for which $w(S_i) > K_s \cdot \bar{S}$. One can use the induction hypothesis because the value of K_s has certainly been reduced by one. ■

4.3. Special limits of server weights

In this subsection, we examine a special case where points are sparse but strongly clustered on a wide range of scales. First, it is demonstrated on two examples where bin weights follow a power law distribution and the resolution number n is large enough. Thereafter, the value of the constant c of Eq. 4.1 is investigated using Alg. 1 and some theoretical results are presented about the Zipf–Mandelbrot law.

4.3.1. Fitting the Zipf–Mandelbrot law to empirical data

The Zipf–Mandelbrot law (Z-M law) [19] is pervasive in a variety of scientific fields, including linguistics, bibliometrics, ecology, biology etc. For a more complete list, we refer the reader to [14]. The probability mass function (PMF) of Z-M law is defined by the following formula if its domain spreads over all positive integers:

$$f(k; \alpha, \beta) = \frac{1}{\zeta(\alpha, 1 + \beta) \cdot (k + \beta)^{\alpha}}, \tag{4.22}$$

where $\zeta(\alpha, 1 + \beta)$ in the denominator is the Hurwitz zeta function.

A numerical maximum log-likelihood method, which is based on the L-BFGS-B [5] implementation of SciPy [15], is used to fit the PMF $f(k; \alpha, \beta)$ to bin weights of a subset of a cosmological N-body simulation (see the left panel of Fig. 1) and intersections of road network of central Illinois taken from the OpenStreetMap website[†] (see the right panel of Fig. 1). Tab. 1 and Fig. 2 show the results of fitting at various resolutions.

[†] http://www.openstreetmap.org

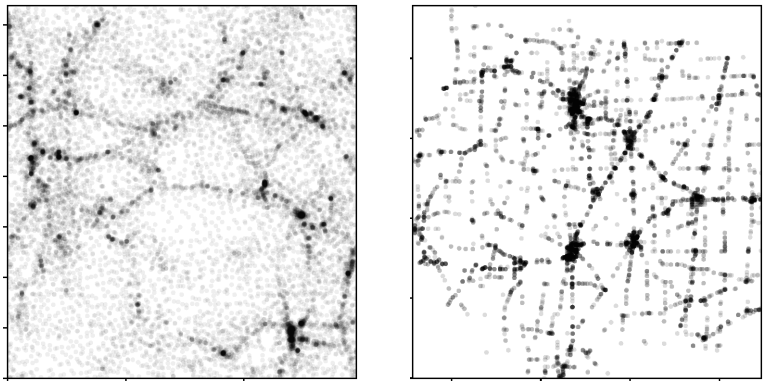

Figure 1. Left: A subset of a cosmological N-body simulation projected to two dimensions. Right: Intersections of the road network of central Illinois.

	N-body simulation				road intersections			
n	α	β	MSE	$x_{\min}$	α	β	MSE	$x_{\min}$
50	4.408	13.053	3.993	0.514	2.985	6.661	3.501	0.517
100	3.386	2.117	2.251	0.551	3.472	5.433	1.857	0.524
200	3.125	0.492	1.280	0.602	4.388	5.498	1.363	0.530
500	3.708	0.192	1.124	0.644	7.857	8.130	0.490	0.537
1000	4.189	0.995	0.492	0.681	11.670	9.983	0.569	0.546

Table 1. Fitting the Z-M law PMF to bin weights of N-body simulation data and road intersections with various resolutions. Mean squared errors (MSE) of maximum log-likelihood estimates are indicated in this table. The values of parameter $x_{\min}$ are also provided for continuous Z-M law fitting. These are computed from the discrete variants with fixed α and β by using Euclidean distance minimization which is based on the L-BFGS-B method.

4.3.2. Estimating the limits of server weights using Alg. 1

It is useful to replace the discrete Z-M law by a continuous one with the following probability density function (PDF) for simpler analytic handling:

$$f(x;\alpha,\beta,x_{\min}) = \frac{(\alpha-1)(x_{\min}+\beta)^{\alpha-1}}{(x+\beta)^{\alpha}} \tag{4.23}$$

where the parameter $x_{\min}$ is introduced as a free parameter with constraints $x \geq x_{\min}$ and $0 < x_{\min} \leq 1$ in order to fit the PDF to the PMF better. As a free parameter, $x_{\min}$ carries no additional meaning.

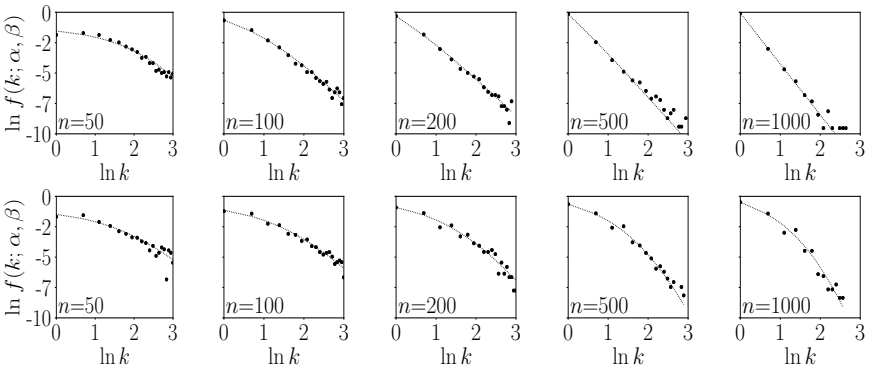

Figure 2. Fitting the Z-M law PMF to bin weights of N-body simulation data (top panel) and road intersections (bottom panel) with various resolutions. On log-log plots, the experimental data are represented by circles and the fitted Z-M PMFs are depicted by dotted lines.

The cumulative distribution function (CDF) is given by the following formula:

$$F(t;\alpha,\beta,x_{\min}) = 1 - \left(\frac{x_{min}+\beta}{t+\beta}\right)^{\alpha-1} \tag{4.24}$$

and the quantile function can be deduced as:

$$Q(p;\alpha,\beta,x_{\min}) = \frac{x_{min}+\beta}{\sqrt[\alpha-1]{1-p}} - \beta. \tag{4.25}$$

Note, that for simplicity of notation, we omit the parameters $\alpha, \beta, x_{\min}$ if it does not lead to misunderstanding.

In the remainder of this section we assume $\alpha > 2$. Alg. 1 determines 'bin weight' intervals, provided that $0 \leq w(B_1) \leq \cdots \leq w\left(B_{n^{\mathcal{D}}}\right)$:

$$\left[w\left(B_{(i-1)\cdot s+1}\right), w\left(B_{i\cdot s}\right)\right] \text{ for all } i = 1, \ldots, \left\lfloor n^{\mathcal{D}}/s \right\rfloor \text{ and}$$
$$\left[w\left(B_{\lfloor n^{\mathcal{D}}/s \rfloor \cdot s+1}\right), w\left(B_{n^{\mathcal{D}}}\right)\right] \text{ if } s \nmid n^{\mathcal{D}}$$

from where each server collects exactly one bin. Our aim is to estimate the limits of server weights, therefore we want to approximate the sum of the start and end of these intervals. Bins with zero weights can be removed for these estimations, so let us denote by b_p the number of bins which have positive weights. It is convenient to assume $s \,|\, b_p$. Although there is loss of generality in assuming divisibility, this assumption makes it easier to understand the method. The treatment of the general case is left to the reader. Let $0 < w(B_1) \leq \cdots \leq w\left(B_{b_p}\right)$, unless otherwise stated. Thus, 'bin weight'

intervals are $[w\left(B_{(i-1)\cdot s+1}\right), w\left(B_{i\cdot s}\right)]$ for all $i = 1, \ldots, b_p/s$. In order to estimate the endpoints of the intervals, one can treat bin weights as realizations of a random variable X which follows the (continuous) Z-M law and its quantile function can be used in the following way. It is clear that the size of the set of all bin weights which are less than the start of the i^{th} interval with $w\left(B_{(i-1)\cdot s+1}\right)$, is $(i-1)\cdot s$. Furthermore, the empirical probability of $X < w\left(B_{(i-1)\cdot s+1}\right)$ is $(i-1)\cdot s/b_p$, therefore $F\left(w\left(B_{(i-1)\cdot s+1}\right)\right) \approx (i-1)\cdot s/b_p$ which implies $Q\left((i-1)\cdot s/b_p\right) \approx w\left(B_{(i-1)\cdot s+1}\right)$.

Let T_i denote $Q\left(((i-1)\cdot s)/b_p\right)$ $(i = 1, \ldots, b_p/s)$. It follows from the above discussion that a lower bound of server weights can be estimated by

$$\sum_{i=1}^{b_p/s} w\left(B_{(i-1)\cdot s+1}\right) \approx \sum_{i=1}^{b_p/s} T_i. \tag{4.26}$$

By definition of the quantile function:

$$\begin{aligned}
\sum_{i=1}^{b_p/s} T_i &= (x_{\min}+\beta)\cdot \sum_{i=0}^{b_p/s-1} \frac{1}{\sqrt[\alpha-1]{1-\frac{i\cdot s}{b_p}}} - \frac{b_p\cdot\beta}{s} > \\
&> (x_{\min}+\beta)\cdot \int_{-1}^{b_p/s-1} \frac{1}{\sqrt[\alpha-1]{1-\frac{x\cdot s}{b_p}}}\mathrm{d}x - \frac{b_p\cdot\beta}{s} = \\
&= (x_{\min}+\beta)\cdot\frac{\alpha-1}{\alpha-2}\cdot\left(\frac{b_p}{s}\cdot\left(1+\frac{s}{b_p}\right)^{\frac{\alpha-2}{\alpha-1}} - \left(\frac{s}{b_p}\right)^{\frac{1}{1-\alpha}}\right) - \frac{b_p\cdot\beta}{s}.
\end{aligned}$$

The sum is treated as the right Riemann sum for the above integral and this is an overestimation because the integrand is monotonically increasing (with respect to x). It follows that

$$w(S_i) \gtrsim (x_{\min}+\beta)\cdot\frac{\alpha-1}{\alpha-2}\cdot\left(\frac{b_p}{s}\cdot\left(1+\frac{s}{b_p}\right)^{\frac{\alpha-2}{\alpha-1}} - \left(\frac{s}{b_p}\right)^{\frac{1}{1-\alpha}}\right) - \frac{b_p\cdot\beta}{s} \tag{4.27}$$

for all $i = 1, \ldots, s$. We denote this lower bound briefly by LB. It is important to note $LB > 0$ because LB is monotonically decreasing with respect to α and it is easy to check that $LB \to (x_{\min}\cdot b_p)/s > 0$ as $\alpha \to \infty$.

Now we focus on the estimation of an upper bound of server weights which is more complicated. Since the range of the random variable X is unbounded, we cannot directly estimate $w\left(B_{b_p}\right)$ but we can choose a threshold T so that bin weights lie under T with a high degree of likelihood ($\geq 99\%$). $T_{b_p/s}$ is the

right choice for T if $b_p > 100 \cdot s$ because

$$\frac{b_p}{s} > 100,$$

$$(x_{\min} + \beta) \cdot \left(\frac{b_p}{s}\right)^{\frac{1}{\alpha-1}} - \beta > (x_{\min} + \beta) \cdot 100^{\frac{1}{\alpha-1}} - \beta,$$

$$T_{b_p/s} > Q(0.99).$$

Then we treat the following sum as the left Riemann sum for the next integral similarly to the previous analysis:

$$\sum_{i=2}^{b_p/s} T_i = (x_{\min} + \beta) \cdot \sum_{i=1}^{b_p/s-1} \frac{1}{\sqrt[\alpha-1]{1 - \frac{i \cdot s}{b_p}}} - \frac{b_p \cdot \beta}{s} <$$

$$< (x_{\min} + \beta) \cdot \int_1^{b_p/s} \frac{1}{\sqrt[\alpha-1]{1 - \frac{x \cdot s}{b_p}}} \mathrm{d}x - \frac{b_p \cdot \beta}{s} =$$

$$= (x_{\min} + \beta) \cdot \frac{\alpha - 1}{\alpha - 2} \cdot \left(\frac{b_p}{s}\right)^{\frac{1}{\alpha-1}} \cdot \left(\frac{b_p}{s} - 1\right)^{\frac{\alpha-2}{\alpha-1}} - \frac{b_p \cdot \beta}{s}.$$

If $b_p > 100 \cdot s$, it follows from the above results that an upper bound of server weights can be estimated by the formula

$$w(S_i) \lesssim (x_{\min} + \beta) \cdot \frac{\alpha - 1}{\alpha - 2} \cdot \left(\frac{b_p}{s}\right)^{\frac{1}{\alpha-1}} \cdot \left(\frac{b_p}{s} - 1\right)^{\frac{\alpha-2}{\alpha-1}} - \frac{b_p \cdot \beta}{s} \tag{4.28}$$

for all $i = 1, \ldots, s$. We denote this upper bound briefly by UB.

Suppose $b_p > 100 \cdot s$. We want to find a constant c_{est} for the Eq. 4.1. Using LB, it follows that

$$\bar{S} - w(S_i) \lesssim \bar{S} - LB \leq c_{\text{est}} \cdot \bar{S},$$
$$1 - LB/\bar{S} \leq c_{\text{est}}$$

and using UB, we have

$$w(S_i) - \bar{S} \lesssim UB - \bar{S} \leq c_{\text{est}} \cdot \bar{S},$$
$$UB/\bar{S} - 1 \leq c_{\text{est}}.$$

Therefore let c_{est} be $\max(1 - LB/\bar{S}, UB/\bar{S} - 1)$. Note: $1 - LB/\bar{S} < 1$ because of $LB > 0$ so the question is whether $UB/\bar{S} - 1$ is less than 1. It is illustrated through concrete examples where $c_{\text{est}} < 1$. Tab. 2 shows it with fixed server number $s = 10$. The experimental constant c_{exp} refers to the ratio of maximum absolute deviation of server weights to mean weight of servers when Alg. 1 is applied for certain bin assignments.

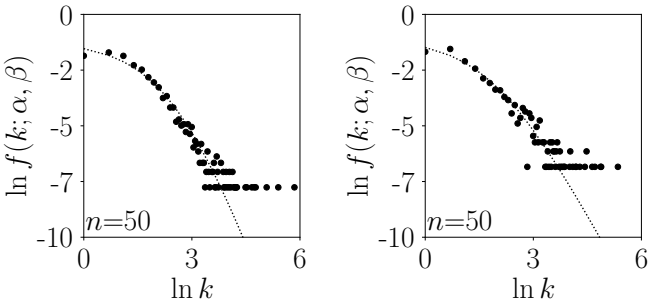

Figure 3. Fitting the Z-M law PMF to bin weights of N-body simulation data (left panel) and road intersections (right panel) with histogram resolution $n = 50$. The distributions of experimental data deviate at higher bin weights from the Z-M law.

	N-body simulation			road intersections		
n	b_p	c_{est}	c_{exp}	b_p	c_{est}	c_{exp}
50	2347	0.094	0.158	935	0.162	0.187
100	5784	0.089	0.070	1569	0.089	0.083
200	9066	0.083	0.034	2322	0.052	0.034
500	12204	0.091	0.013	3463	0.037	0.012
1000	13748	0.322	0.005	4235	0.038	0.009

Table 2. Comparison of estimated and experimental constants for Eq. 4.1 when server number s is fixed to 10. These constants are less than 1 for all resolution numbers. The value of the constant c_{est} is an overestimate of the experimental one except for $n = 50$. The distributions of experimental data deviate at higher bin weights from the Z-M law and, for resolution number $n = 50$, higher bin weights are significant outliers compared to lower bin weights (see Fig. 3). The numbers of bins with positive weights (b_p) are also indicated and it can be seen that the assumption $b_p > 100 \cdot s$ is met almost everywhere which is important for the estimation (see details in the text). The corresponding fitted parameters (α, β, $x_{\min}$) for histogram resolution n are contained in Tab. 1.

5. Conclusions

Our earlier work presented the SG17 algorithm. In this paper we investigated how one can accelerate this algorithm by using upper limits of the absolute deviation of server weights, for which we derived formulas. A special case was examined where data points are sparse but strongly clustered on a wide range of scales. It was demonstrated that if histogram resolution is large enough, the related bin weights follow the Zipf–Mandelbrot law. Therefore, some related theoretical results were derived about how much the upper limit of absolute deviation of server weights can be improved in this special case. Among others, our future work focus will be on cases when data points are distributed differently. For example, if data points are more or less uniformly distributed, bin weights follow a Poisson distribution with mean (rate) at the average weight of bins.

We mentioned that one can use a generic two-tier index scheme including global and local index structures in the storage layer where the global index structure is built on the multidimensional histogram and is maintained in the central server. However, there are additional interesting questions about this approach: how can one

- ... maintain this architecture when data are updated (if static, non-time-varying condition is relaxed for data),
- ... replicate data between servers to ensure fault tolerance etc.

Author contributions: All authors contributed to discussion and interpretation of the results. J. M. Szalai-Gindl conducted the literature review and the analysis, contributed to the paper's concept, and wrote the paper. A. Kiss contributed to the paper's concept and critical revision of the manuscript. L. Dobos contributed to the paper's concept, coordinated data collection and critical revision of the manuscript.

References

[1] **Aji, A.,** *High Performance Spatial Query Processing for Large Scale Spatial Data Warehousing*, PhD thesis, Emory University, 2014.

[2] **Aluru, S. and F.E. Sevilgen,** Parallel domain decomposition and load balancing using space-filling curves, *Fourth International Conference on High-Performance Computing* (Bangalore, India, 18-21 December, 1997), Proceedings Fourth International Conference on High-Performance Computing, IEEE, Bangalore, India, 1997, 230–235.

[3] **An, N., R. Lu, L. Qian, A. Sivasubramaniam and T. Keefe,** Storing spatial data on a network of workstations, *Cluster Computing*, **2(4)** (1999), 259–270.

[4] **Bhatia, R., R.K. Sinha and C.-M. Chen,** A hierarchical technique for constructing efficient declustering schemes for range queries, *The Computer Journal*, **46(4)** (2003), 358–377.

[5] **Byrd, R.H., P. Lu, J. Nocedal and C. Zhu,** A limited memory algorithm for bound constrained optimization, *SIAM Journal on Scientific Computing*, **16(5)** (1995), 1190–1208.

[6] **Chakrabarti, K. and S. Mehrotra,** The hybrid tree: An index structure for high dimensional feature spaces, *15th International Conference on Data Engineering* (Sydney, Australia, March 23-26, 1999), Proceedings of the 15th International Conference on Data Engineering, IEEE, Sydney, Australia, 1999, 440–447.

[7] **Charnes, A., W.W. Cooper and R.O. Ferguson,** Optimal estimation of executive compensation by linear programming, *Management science*, **1(2)** (1955), 138–151.

[8] **Chen, C.-M. and C.T. Cheng,** From discrepancy to declustering: near-optimal multidimensional declustering strategies for range queries, *Journal of the ACM (JACM)*, **51(1)** (2004), 46–73.

[9] **Choi, H.-H., M.-Y. Lee and K.-C. Lee,** Distributed high dimensional indexing for k-NN search, *The Journal of Supercomputing*, **62(3)** (2012), 1362–1384.

[10] **Csabai, I., L. Dobos, A. Kiss and J.M. Szalai-Gindl,** Some mathematical properties of the performance measures applied for point cloud databases, *Annales Univ. Sci. Budapest., Sect. Comp.*, **47** (2018), 197–209.

[11] **Eldawy, A. and M.F. Mokbel,** SpatialHadoop: A MapReduce framework for spatial data in: J. Gehrke, W. Lehner, K. Shim, S. K. Cha, G. M. Lohman (Ed.) *31st IEEE International Conference on Data Engineering, ICDE 2015* (Seoul, South Korea, April 13–17, 2015), IEEE, Seoul, South Korea, 2015, 1352–1363.

[12] **Golab, L., M. Hadjieleftheriou, H. Karloff and B. Saha,** Distributed Data Placement via Graph Partitioning, https://arxiv.org/pdf/1312.0285.pdf, (2013).

[13] **Gong, J., Q. Zhu, R. Zhong, Y. Zhang and X. Xie,** An efficient point cloud management method based on a 3D R-tree, *Photogrammetric Engineering & Remote Sensing*, **78(4)** (2012), 373–381.

[14] **Izsák, J.,** Some practical aspects of fitting and testing the Zipf–Mandelbrot model: A short essay, *Scientometrics*, **67(1)** (2006), 107–120.

[15] **Jones, E., T. Oliphant, P. Peterson and others,** {SciPy}: Open source scientific tools for {Python}, (2001–), http://www.scipy.org/, [Online; accessed 2019-06-02].

[16] **Kamel, I. and C. Faloutsos,** Parallel R-trees, in: Michael Stonebraker (Ed.) *1992 ACM SIGMOD International Conference on Management of Data* (San Diego, California, USA, June 2–5, 1992), Proceedings of the 1992 ACM SIGMOD International Conference on Management of Data, ACM, San Diego, California, USA, 1992, 195–204.

[17] **Koudas, N., C. Faloutsos and I. Kamel,** Declustering spatial databases on a multi-computer architecture, *Advances in Database Technology - EDBT'96, 5th International Conference on Extending Database Technology* (Avignon, France, March 25–29, 1996), Springer, Avignon, France, 1996, 592–614.

[18] **Koyutürk, M. and C. Aykanat,** Iterative-improvement-based declustering heuristics for multi-disk databases, *Information Systems*, **30(1)** (2005), 47–70.

[19] **Mandelbrot, B.,** An informational theory of the statistical structure of language, *Communication Theory*, **84** (1953), 486–502.

[20] **Moon, B., A. Acharya and J. Saltz,** Study of scalable declustering algorithms for parallel grid files, *10th International Parallel Processing Symposium* (Honolulu, Hawaii, USA, April 15-19, 1996), Proceedings of IPPS '96, IEEE, Honolulu, Hawaii, USA, 1996, 434–440.

[21] **Nishimura, S., S. Das, D. Agrawal and A. El Abbadi,** MD-HBase: A scalable multi-dimensional data infrastructure for location aware services, *12th IEEE International Conference on Mobile Data Management* (Luleå, Sweden, June 6-9, 2011, Volume 1), IEEE, Luleå, Sweden, 2011, 7–16.

[22] **Papadopoulos, A. and Y. Manolopoulos,** Parallel bulk-loading of spatial data, *Parallel Computing*, **29(10)** (2003), 1419–1444.

[23] **Patwary, M.M.A., N.R. Satish, N. Sundaram, J. Liu, P. Sadowski, E. Racah, S. Byna, C. Tull, W. Bhimji, P. Dubey and others,** PANDA: Extreme scale parallel K-nearest neighbor on distributed architectures, *2016 IEEE International Parallel and Distributed Processing Symposium, IPDPS 2016* (Chicago, IL, USA, May 23–27, 2016), IEEE, Chicago, IL, USA, 2016, 494–503.

[24] **Samet, H.,** *Foundations of Multidimensional and Metric Data Structures*, Chapter 2, p. 271, Morgan Kaufmann, 2006.

[25] **Schnitzer, B. and S.T. Leutenegger,** Master-client *R*-trees: A new parallel *R*-tree architecture, *Eleventh International Conference on Scientific and Statistical Database Management'99* (Cleveland, OH, USA, July 28-30, 1999), Proceedings of the Eleventh International Conference on Scientific and Statistical Database Management, IEEE, Cleveland, OH, USA, 1999, 68–77.

[26] **Szalai-Gindl, J.M., L. Dobos and I. Csabai,** Tiling strategies for distributed point cloud databases, in: A. Choudhary, K. Wu, F. Rusu, G. Trajcevski, A. Shoshani, B. Dong, B. Zhang (Ed.) *Conference on Scientific and Statistical Database Management, SSDBM '17* (Chicago, IL, USA, June 27–29, 2017), Proceedings of the 29th International Conference on Scientific and Statistical Database Management, ACM, New York, 2017, 32:1–32:6.

[27] **Tirthapura, S., S. Seal and S. Aluru,** A formal analysis of space filling curves for parallel domain decomposition, *2006 International Conference on Parallel Processing* (Columbus, Ohio, USA, August 14-18, 2006), IEEE, Columbus, Ohio, USA, 2006, 505–512.

[28] **Van, L.H. and A. Takasu,** An efficient distributed index for geospatial databases, *26th International Conference on Database and Expert Systems Applications-Volume 9261* (Valencia, Spain, September 1-4, 2015), Proceedings, Part I, of the 26th International Conference on Database and Expert Systems Applications, Springer, Valencia, Spain, 2015, 28–42.

[29] **Vo, H., A. Aji and F. Wang,** SATO: a spatial data partitioning framework for scalable query processing, *22nd ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems* (Dallas/Fort Worth, TX, USA, November 4–7, 2014), Proceedings of the 22nd ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, ACM, Dallas/Fort Worth, TX, USA, 2014, 545–548.

[30] **Wagner, H.M.,** Linear programming techniques for regression analysis, *Journal of the American Statistical Association*, **54(285)** (1959), 206–212.

[31] **Yang, Y., L. Wu, J. Guo and S. Liu,** Research on distributed Hilbert *R* tree spatial index based on BIRCH clustering, *The 20th International Conference on Geoinformatics (GEOINFORMATICS)* (Hong Kong, China, June 15–17, 2012), IEEE, Hong Kong, China, 2012, 1–5.

[32] **You, S.,** *Large-Scale Spatial Data Management on Modern Parallel and Distributed Platforms*, PhD thesis, City University of New York, 2016.

[33] **Zhong, Y., J. Han, T. Zhang, Z. Li, J. Fang and G. Chen,** Towards parallel spatial query processing for big spatial data, *26th IEEE International Parallel and Distributed Processing Symposium Workshops & PhD Forum, IPDPS 2012* (Shanghai, China, May 21–25, 2012), IEEE, Shanghai, China, 2012, 2085–2094.

I. Csabai and L. Dobos
Department of Physics of Complex Systems
Eötvös Loránd University
Budapest
Hungary
csabai@complex.elte.hu
dobos@complex.elte.hu

A. Kiss and J. M. Szalai-Gindl
Department of Information Systems
Eötvös Loránd University
Budapest
Hungary
kiss@inf.elte.hu
szalaigindl@inf.elte.hu